Syntatic Error:

```
PS C:\Users\nihal\VSCode\COE2SH4\Labs\Lab2\lab2-inela0> make
gcc -c -o Question4.o Question4.c -I. -lm -g
Question4.c: In function 'sortDatabyBubble':
Question4.c:44:17: error: expected ';' before 'array'
                array[i + 1].charData = array[i].charData;
                ^~~~~
make.exe": *** [Question4.o] Error 1
PS C:\Users\nihal\VSCode\COE2SH4\Labs\Lab2\lab2-inela0>
```

After adding a semicolon to line 43, the program runs without any compiler errors.

Symentic Errors:

1. Within the for loop condition statement, i is set to be smaller or equal to size-1, whereas in order to go through all data, size-2 is enough.

   Process:

   ```
   This GDB was configured as "mingw32".
   For bug reporting instructions, please see:
   <http://www.gnu.org/software/gdb/bugs/>...
   Reading symbols from C:\Users\nihal\VSCode\COE2SH4\Labs\Lab2\lab2-inela0\Lab2.exe...done.
   (gdb) run
   Starting program: C:\Users\nihal\VSCode\COE2SH4\Labs\Lab2\lab2-inela0/./Lab2.exe
   [New Thread 36548.0x9244]
   [New Thread 36548.0x3bc4]

   Program received signal SIGSEGV, Segmentation fault.
   0x00402e04 in TestQ4_BubbleSort_1 (tc=0x6d2690) at testCases.c:351
   351                     CuAssertIntEquals(tc, expected[i].intData, input[i].intData);
   (gdb)
   ```

   Segmentation fault means the progrsm is trying to access a memory location that is not allowed to access, meaning that we are not trying to access an index that is out of bounds. Thus, we should change the for loop condition statement so that the last index we access is at size-2

   ```
   (gdb) run
   Starting program: C:\Users\nihal\VSCode\COE2SH4\Labs\Lab2\lab2-inela0/./Lab2.exe
   [New Thread 30376.0x767c]
   [New Thread 30376.0x2078]
   F

   There was 1 failure:
   1) TestQ4_BubbleSort_1: testCases.c:352: expected <122> but was <97>

   !!!FAILURES!!!
   Runs: 1 Passes: 0 Fails: 1

   [Inferior 1 (process 30376) exited normally]
   ```

   After we fix the error, we no longer get the segmentation fault warning :)

2. Inside the if statement,
    a. first, temp values are set equal to the current data,
    b. then, the current is set back to temp - which reverses the first step and defeats purpose of creating a temp variable
    c. Finally, sets next data equal to current - this basically means that we are creating an array that has the exact same value in each index, which is the value at index 0

In order for the bubble sort to work correctly, we must

a) First, set temp to next
b) Then, set next to current
c) Finally, set current to temp

Process:

Running the debugger for the first test case

```
//=========================================================
//================Question 4===============================
void TestQ4_BubbleSort_1(CuTest *tc)
{
    int n=6;
    struct Q4Struct input[]={{10, 'c'}, {2, 'B'}, {-5, 'k'}, {12, 'z'}, {77, 'a'}, {-42, '?'}};
    struct Q4Struct expected[]={{-42, '?'}, {-5, 'k'}, {2, 'B'}, {10, 'c'}, {12, 'z'}, {77, 'a'}};

    sortDatabyBubble(input, n);

    int i;
    for (i=0; i<n; i++)
    {
        CuAssertIntEquals(tc, expected[i].intData, input[i].intData);
        CuAssertIntEquals(tc, expected[i].charData, input[i].charData);
    }
}
```

```
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = 10, charData = 99 'c'}
i = 1
curr = 10
next = -5
done = 0
(gdb) c
Continuing.

Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = 10, charData = 99 'c'}
i = 2
curr = 10
next = 12
done = 0
(gdb) c
Continuing.
```

```
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = 10, charData = 99 'c'}
i = 3
curr = 12
next = 77
done = 0
(gdb) c
Continuing.

Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = 10, charData = 99 'c'}
i = 4
curr = 77
next = -42
done = 0
(gdb)
```

As can be seen with the info locals feature, temp variable always stays the same, even though it should be. After we implement the changes mentioned above, lets run the debugger again and see the difference.

```
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = 0, charData = 0 '\000'}
i = 0
curr = 10
next = 2
done = 1
(gdb) c
Continuing.

Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = 2, charData = 66 'B'}
i = 1
curr = 10
next = -5
done = 0
(gdb) info locals
temp = {intData = 2, charData = 66 'B'}
```

```
done = 0
(gdb) info locals
temp = {intData = 2, charData = 66 'B'}
i = 1
curr = 10
next = -5
done = 0
(gdb) c
Continuing.

Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
Breakpoint 1, sortDatabyBubble (array=0x61fe18, size=6) at Question4.c:35
35                  if(curr > next)
(gdb) info locals
temp = {intData = -5, charData = 107 'k'}
i = 2
curr = 10
next = 12
done = 0
(gdb)
```

The temp variable is now changing and swap works correctly!